RIPS - A Rapid Pattern Simulator for BRDFs

April, 1999

Herbert L. Hirsch
MTL Systems, Inc.
Dayton, Ohio 45432
hhirsch@mtl.com

ABSTRACT

A methodology for a fast-executing electromagnetic interference pattern modeling algorithm is given. The general issues pertaining to interference pattern modeling are first discussed. The algorithm is then developed and explained in detail, based upon its genesis in radio frequency antenna gain pattern modeling. Finally applications and results for modeling BRDFs are presented. The technique is shown to provide substantial computational reduction over contemporary modeling methods, while maintaining accuracy and precision. Future efforts are expected to provide additional support utilities for the algorithm to create a CAD-like pattern modeling tool.

1.0 INTRODUCTION

Bi-directional reflectivity distribution functions (BRDFs) are a kind of electromagnetic interference pattern, as are radio frequency (RF) antenna gain patterns and many other patterns which describe the behavior of an electromagnetic wave when driven through an aperture or encountering an obstacle. The need to model and simulate such patterns occurs frequently in the course of infrared countermeasure (IRCM) or counter-countermeasure (IRCCM) development, simulation, and testing. In such instances, the sheer complexity of the mathematics of these electromagnetic phenomena can render the simulation very slow, often too slow to support real-time or near-real-time modeling needs. In this paper, we describe a fast-executing algorithm by which BRDFs and other such interference patterns may be modeled, without sacrificing necessary accuracy or precision. We begin with an overview of the interference pattern modeling problem in general, then provide an overview of our RIPS modeling algorithm, and conclude with discussions of its application and results from modeling some nominal BRDFs.

Form SF298 Citation Data

Report Date *("DD MON YYYY")* 00041999	**Report Type** N/A	**Dates Covered (from... to)** *("DD MON YYYY")*

Title and Subtitle RIPS - A Rapid Pattern Simulator for BRDFs	**Contract or Grant Number**
	Program Element Number
Authors Hirsch, Herbert L.	**Project Number**
	Task Number
	Work Unit Number
Performing Organization Name(s) and Address(es) MTL Systems, Inc. Dayton, Ohio 45432	**Performing Organization Number(s)**
Sponsoring/Monitoring Agency Name(s) and Address(es)	**Monitoring Agency Acronym**
	Monitoring Agency Report Number(s)
Distribution/Availability Statement Approved for public release, distribution unlimited	
Supplementary Notes	
Abstract	
Subject Terms	
Document Classification unclassified	**Classification of SF298** unclassified
Classification of Abstract unclassified	**Limitation of Abstract** unlimited
Number of Pages 15	

2.0 THE INTERFERENCE PATTERN MODELING PROBLEM

Electromagnetic interference patterns are pervasive in our technical culture. They occur basically any time an electromagnetic wave impacts an obstacle. A common instance is the radio frequency (RF) or radar *antenna gain pattern*, which results from the drive signal exciting the radiating elements of the antenna structure (wire dipoles, horns, or array elements, for example). Another instance is the *bi-directional reflectivity distribution, or BRDF*, which represents the reflected portion of a wave at optical wavelengths after impacting some surface.

We often need to model or simulate these patterns. For example, RF antenna gain and surface reflectivity quantities are paramount to modeling cellular communications systems or military engagements involving multiple airborne and ground-based RF and radar systems. Optical reflectivity quantities are required for simulating infrared (IR) and visual sensing systems. Furthermore, when such simulations are used in conjunction with actual hardware-in-the-loop or humans-in-the-loop, these simulations must execute fast enough to support real-time system operation. There are many other instances as well, but the common denominators among all such patterns are the following:

1. They all appear as multiple-lobed amplitude as a function of observation angle (to some reference axis of the object transmitting or reflecting the wave). The amplitude may be relative gain (dimensionless), radio frequency electrical field strength (volts) or power (watts), optical wavelength reflectivity (dimensionless), or some other measure of magnitude.

2. In all but the simplest cases, they are *computationally-expensive or memory-intensive to model*. The mathematics are complex and involve multiple integrations as well as trigonometric relationships. If they are pre-calculated, storage at high resolution consumes memory and storage at lower resolution requires two-dimensional interpolation between located data points. This makes them difficult to model in a manner which supports real-time execution.

An example of a multiple-lobed interference pattern is shown in Figure 1. This illustrates typical interference pattern behavior, which is a set of amplitude lobes as a function of the angle to some observation point (the *observation angle*). As Figure 1 shows, these patterns typically exhibit a main lobe, and a set of secondary lobes (or sidelobes) around it. The sidelobes are often, but not always, very symmetrical in any given angular direction from the main lobe, and exhibit alternating mathematical sign (+ or -), which is really a phase-reversal property.

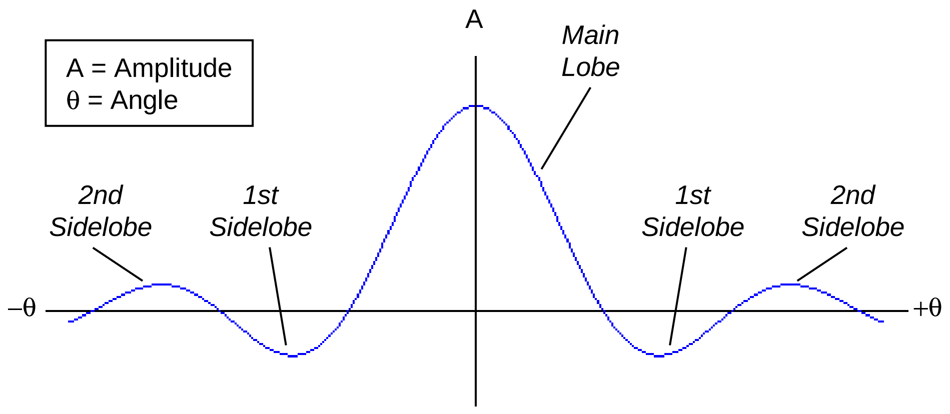


Figure 1. A Simple Pattern - the Sinc Function in Two Dimensions

The pattern of Figure 1 is a simple pattern - the classic *sinc function* ($A = \sin X/X$), where A is the amplitude and X is some function of the observation angle, θ. Hence, we call it a two-dimensional (A,θ) interference pattern. Actually, this sinc function is a legitimate interference pattern, representing the electric field intensity (voltage gain) of a uniform-current strip of length, L, and negligible width, as a function of the angle, θ, between a vector normal to the strip and the observation point. Here, both the vector and observation point lie in a plane orthogonal to the length. The actual formula is:

$$E = \sin(\pi u L/\lambda)/(\pi u L/\lambda) \qquad (1)$$

Where:	E	=	Electric Field Strength
	u	=	$\sin\theta$
	λ	=	wavelength

However, this simple sinc function is actually the solution to an integral expression for the voltage gain, which is:

$$A = \int_L I(z)\, e^{jkuz} dz \qquad (2)$$

Where:	$I(z)$	=	Current along the strip (z-direction)
	k	=	wave number ($2\pi/\lambda$)

In practical application, however, we are dealing with objects of non-negligible width, and we cannot constrain the observation point to a plane strictly orthogonal to some axis of the object. Hence, we have two angular dimensions, typically azimuth (θ) and elevation (α), with respect to a normal vector originating at the object. If we consider the case of a two-dimensional, rectangular aperture excited by an illumination function (as would be the case for certain radar antenna types), the integral equation becomes:

$$A = \int_{-W/2}^{W/2} \int_{-H/2}^{H/2} g(y,z) e^{-j2\pi(yu+zv)} dy dz \qquad (3)$$

Where:	W	=	Aperture width in the y direction
	H	=	Aperture height in the z direction
	u	=	$(1/\lambda)\sin\theta\cos\alpha$
	v	=	$(1/\lambda)\sin\theta\sin\alpha$
	$g(y,z)$	=	The illumination function across the aperture

In such cases, the integral equations for gain can involve complex variables ($a + bi$). Also, the illumination function, $g(y,z)$ may be a complex trigonometric function itself. The gain (A) in any direction (θ,α) would be obtained from solving the integral equation, or numerically integrating if no closed-form solution to the integral was obtainable. The gain pattern, now in three dimensions (A, θ, α), would typically appear as shown in Figure 2. In this example, the pattern is quite symmetrical, with each sidelobe appearing as a maxima or minima "ring" around the main lobe. However, on some patterns the sidelobe rings may exhibit their own maxima and minima, as "peaks" and "valleys" occurring radially around the sidelobe ring.

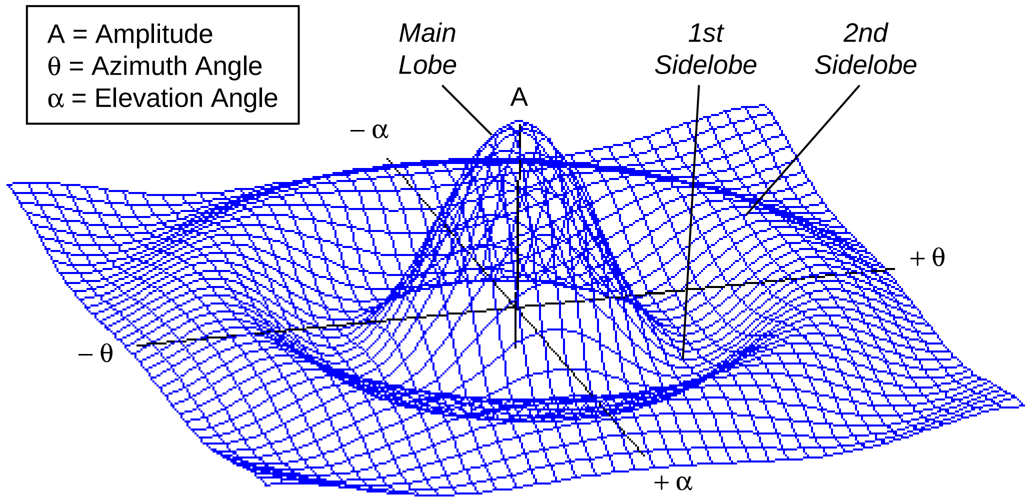


Figure 2. Interference (gain) Pattern in Three Dimensions

We are not going to dwell on electromagnetic theory here. The point is simply that the computational burden generally encountered in modeling these patterns is significant. If we wish to obtain the amplitude (gain) at any particular azimuth-elevation angle combination using a rigorous computation of this phenomenon, we are in for a serious computational load. We must first produce a mathematical expression for the illumination function which impacts or illuminates the object. We must then calculate the distribution of this illumination function over the object's surface, which generally requires a two-dimensional surface integration of complex expressions, and multiply by the surface reflectivity if the situation is reflective. In other words, we are dealing with two-dimensional integrals in complex mathematics - not a computationally trivial situation, and numerical integrations take time. Furthermore, we must perform this computation at each instance of an azimuth-elevation angle change, if we wish to simulate dynamic conditions such as aircraft with sensor systems as they maneuver about. For real-time models such as are used for training, with high pulse-rate radar simulations or fast frame-rate optical sensor models, this can require a large amount of processing upon expensive parallel computing machines. In wavelengths and phenomenology other than RF, such as the BRDF for example, the specific equations are different. However, the level of computational expense, and the additional demands wrought by real-time, dynamic simulation requirements, remain significant. Such effects are inherent to rigorous electromagnetic computations or time-frequency domain transformations through Fourier methods.

Alternatively, we might pre-calculate an entire interference pattern using the rigorous mathematics, and store the values in a lookup table for use upon demand. However, this leads us right into the usual lookup table issue - resolution versus access speed. When, for example, a radar antenna gain or optical reflectivity is used as part of a sensitive angle tracking function, high precision is required. If we index our amplitudes by azimuth and elevation angle, we have a two-dimensional interpolation to perform once we locate the bounding data points in both dimensions in our table. If we want to avoid interpolation, then we must compute and store our data at a very high degree of resolution, which requires more memory capacity and can make the search routines rather slow. Therefore, a lookup table may or may not be an adequate solution. In other words our two modeling alternatives, consisting of rigorous computation or lookup tables, are either computationally-expensive, memory-intensive, or time consuming in search and retrieval of data. Fortunately, there is a third alternative.

3.0 A LOBE-BUILDING MODELING APPROACH

Our approach to modeling these interference patterns is a simple and effective one. Looking back at Figure 2, or at any other pattern of this nature, we note that it is basically an *assimilation of lobes*, or at least can be thought of that way. Even patterns which are not so symmetrical about both angular axes as that of Figure 2 may be basically considered an assemblage of lobes. Patterns which have non-uniform sidelobe levels, exhibiting additional minima and maxima around the sidelobe "rings" are still a system of lobes. Hence, if we have a technique to model these lobes with simple mathematics, while still retaining modeling accuracy, we also have a valuable means to simulate these phenomena while conserving execution time and computing or memory resources. Such a technique to model these patterns is what we describe next.

We originally developed this algorithm, or modeling function, to simulate radar antenna gain patterns [1], and then found applications in BRDFs and other interference pattern phenomenology. Our premise was that if we could find a computationally-simple function which would give us control over the attributes of a lobe, as shown in Figure 3, and would produce lobes that could be fitted together seamlessly in mathematical space to form complete patterns, as shown in Figure 4, then we would have a solution. We basically sought a simple function which would give us sufficient control over the angular boundaries, amplitude, and shape of the rising and falling segments of a given lobe. Sufficient control in this context means the ability to accurately model the shape of the segments and lobes and the ability to make them connect seamlessly, with no anomalous behavior at their intersections. This control is essential to successfully apply an empirically-based technique such as ours.

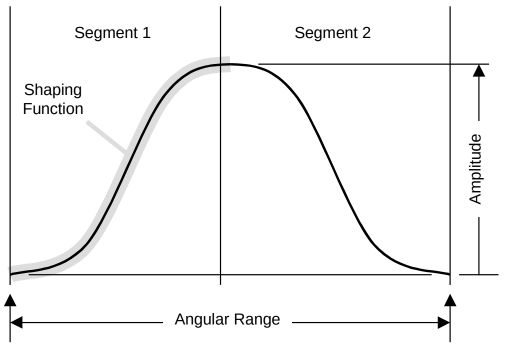


Figure 3.
Lobe Attributes

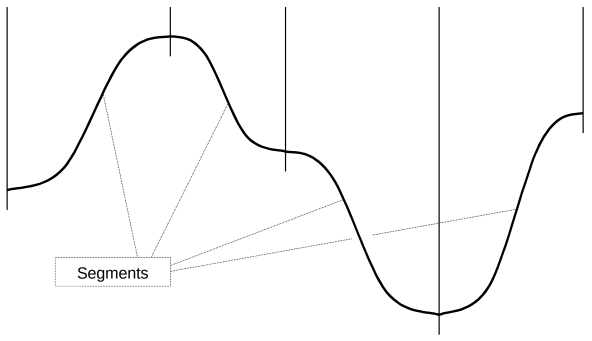


Figure 4.
Fitting Segments into a Pattern

Our algorithm begins with the shaping function - a simple sinusoidal function of the observation angle, taken to an exponent (power). The function actually exists in the logarithmic domain, but we'll take care of that aspect later. Hence, it is simply this:

$$F = (\sin\gamma)^{C} \tag{4}$$

Where: F = The shaping function
γ = $\cos^{-1}(\cos\theta\cos\alpha)$:the composite azimuth-elevation angle
C = a shaping exponent

Now, since this shaping function is to be continuous over only a segment or a lobe, we need to normalize the look angle to the segment or lobe angular boundaries. We can accomplish this as:

$$F = \{\sin[\pi(\gamma-\gamma_L)/(\gamma_U-\gamma_L)]\}^C \quad (5)$$

Where: γ_L = The lower angular boundary of the lobe
γ_U = The upper angular boundary of the lobe

The function of *Eq. 5* produces a complete lobe, as shown previously in Figure 3, consisting of both the lower and upper segments (Segments 1 and 2), in a continuous function. In most cases lobes are quite symmetrical and this works perfectly well, but if we wish to have a different shape for each segment of the lobe, then we simply divide this expression into two separate ones for each segment. For our discussions here, we will use the complete lobe, as we have found this to suffice for most practical applications.

At this point, the function simply exists between 0 and 1.0, the range of a sinusoidal function. To apply it to a particular range of amplitude values, we need to add amplitude range and bias terms. We do this simply by multiplying the expression of Eq. 5 by the range and subtracting the amplitude minimum, which is accomplished as:

$$F = \{\sin[\pi(\gamma-\gamma_L)/(\gamma_U-\gamma_L)]\}^C(B_U-B_L) - B_L \quad (6)$$

Where: B_U = The upper amplitude boundary (maximum) of the lobe
B_L = The lower amplitude boundary (minimum) of the lobe

As with the shaping coefficient, this imparts the same amplitude characteristics across both segments of a lobe. As before, if we wish to have a different amplitude range for each segment of the lobe, then we simply need to apply a different minimum amplitude (B_L) for each segment, since both segments must have the same maximum amplitude where they meet.

Our final step is to get this function out of the logarithmic domain. We do this simply by taking 10.0 to a power represented by the expression of *Eq. 6.* This gets us a decimal expression for the lobe amplitude, A, as the pair of equations:

$$F = \{\sin[\pi(\gamma-\gamma_L)/(\gamma_U-\gamma_L)]\}^C(B_U-B_L) + B_L \text{ \textit{(from Eq. 6)}, and} \quad (7a)$$

$$A = 10^F \quad (7b)$$

At this point, it is interesting to explore the control we have over the shape of the lobe with the shaping exponent, C. Figure 5 illustrates this for several values of C, showing that we may not only control the width of the lobe, but we may also control the number of inflection points our lobe has. As this figure shows, higher values of C not only narrow the function but also cause it to change from approaching the baseline orthogonally to approaching it tangentially. This is tremendously powerful in fitting lobes together seamlessly and is obtained without the use of higher-ordered polynomials, which can become computationally demanding and often result in a lack of intuitive control by the user (who cannot easily keep track of the interactive effects of the higher-ordered coefficients upon the shape of the function).

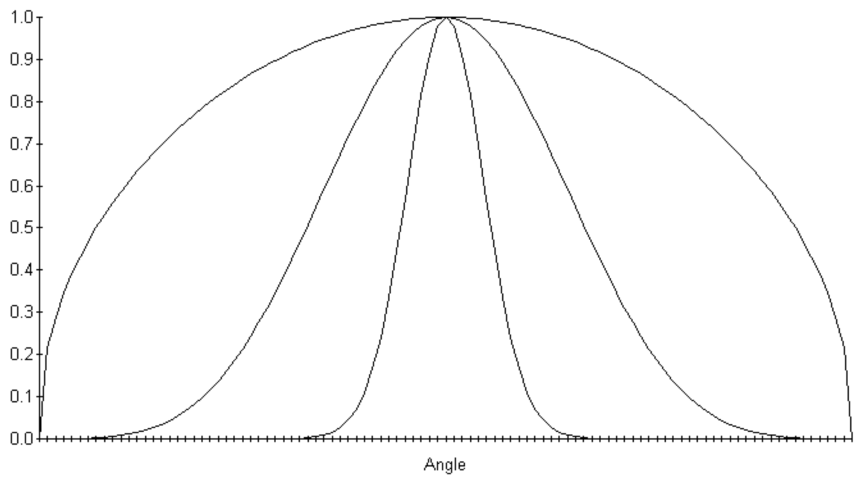


Figure 5. Lobe Shape Control with C, the Shaping Exponent

In practice, we found the expressions of *Eqs. 7a and 7b* were all we needed to simulate the complex behavior of many antenna gain patterns. However, we made two minor alterations. First, since the practice of antenna theory generally operates in decibel (dB) notation, we tailored Eq. 7a to use amplitude dimensions expressed in a quantity we called dB Voltage gain, where dB voltage gain is 10F. This was done as a matter of convention and convenience, and led us to use the following quantities for the upper and lower amplitude boundaries:

B_U = M = Maximum lobe amplitude with respect to peak (a negative dB voltage gain number)
B_L = -100 (in dB voltage gain)

Which slightly altered these expressions to:

$$F = \{\sin[\pi(\gamma-\gamma_L)/(\gamma_U-\gamma_L)]\}^C(M+100) - 100 \quad \textit{(from Eq. 6)}, \text{ and} \qquad (8a)$$
$$A = 10^{F/10} \qquad (8b)$$

Second, although this function performed adequately in its present form, we found that we could obtain a near-perfect fit to patterns we were modeling by adding the ability to make the function a bit more "pointed" at the top. Here, we defined a simple triangular shape function, T, which produces a linear value from the minimum amplitude value at the upper or lower angular boundaries to the maximum amplitude value at the center of the lobe, γ_C. It is simply a triangle whose base is the lobe's angular width and whose height is the same amplitude range as was defined for the basic lobe shaping function.

$$T = \{1 - (\gamma-\gamma_C)/(\gamma_L-\gamma_C)\}F_A \text{ for the lower segment, or} \qquad (9a)$$
$$T = \{1 - (\gamma-\gamma_C)/(\gamma_U-\gamma_C)\}F_A \text{ for the upper segment} \qquad (9b)$$

Where: $F_A = 10^{[(M+100) - 100]/10}$ (an amplitude conversion factor)
γ_C = The center angle of the lobe (half the distance between γ_L and γ_U, if the lobe is a symmetrical one)

In cases where the lobe has symmetry across both upper and lower segments, *Eqs. 9a and 9b* produce identical results. The amplitude conversion factor, F_A, simply uses the same amplitude range as was defined in *Eq. 8a*, takes it out of the logarithmic domain, and applies it to the triangular function, T.

This function is very computationally inexpensive since the amplitude conversion factor remains unchanged for a lobe or lobe segment, and may simply be calculated once for each lobe or segment and stored. The angular variables are the only dynamic ones.

To apply our triangular function, we simply define a weighting coefficient, W, which regulates the relative contribution of T, the triangular function, and F, the basic shape function we defined earlier. This results in a modification of *Eq. 8b* to:

$$A = (10^{F/10} + WT)/(W + 1) \qquad (10)$$

Summarizing, our model, consists of *Eqs. 8a, 9a, 9b, and 10,* representing a computationally-simple way to model these complex electromagnetic interference patterns. We have, in fact, applied it to modeling many interference patterns in both the radio-frequency and optical spectral regions. In these utilizations, we simply stored a set of coefficients (γ_L γ_U, C, B_L, B_U, and W) for each lobe, using the angular boundaries (γ_L γ_U) in our database search scheme. Then, for each instance of some observation angle, γ, our scheme simply located the angular boundaries within which the observation angle occurred, and applied the located set of coefficients to these expressions. Since the quantization of data sets in the database was at the lobe-level, this represented significantly fewer search locations than would be the case for a whole pattern stored at a reasonable resolution, say of a degree or less. Since the expressions themselves were orders of magnitude less time-consuming than rigorous algorithms, the results were obtained much faster than such rigorous computations. Hence, we won on all accounts: (1) fewer data locations to search than an interpolative lookup table, and (2) a fast-executing algorithm virtually as simple as interpolation mathematics. Next, we look at how this model works in a few example cases.

4.0 BRDF MODELING APPLICATIONS AND RESULTS

The basic process of applying this model to simulating some interference pattern is to first obtain data representing the pattern to be modeled, and then to "fit" the model to the pattern, using the coefficients discussed earlier to obtain the lobe characteristics which achieve the fit. Once the coefficients are determined and stored, they may be retrieved and applied to the various lobes of the pattern as necessary, for real-time simulation or other, less-demanding purposes as well. To perform this fitting process, we developed a simple, Visual Basic program we called ANTPAT (Antenna Pattern fitting tool - since it was primarily antenna patterns we were dealing with at the time) which permits us to display data for an interference pattern we wish to model, and then adjust our coefficients to obtain the fit we desire. It is a 2-dimensional model, and requires operation across both angular dimensions of a pattern to get a complete, 3-dimensional fit. However, when the interference patterns are symmetrical, as is often the case, the 2-dimensional fit satisfies the additional angular dimension as well. Next, we illustrate how the fitting process works for a BRDF, which represents the reflectivity of a surface as a function of the observation angle with respect to the incidence angle.

To begin, let's look at an actual BRDF we want to model, shown in Figure 6. This is the waveform display screen of ANTPAT, which displays both the pattern to be fitted as the "Overlay Plot" in green and the results of our fitting process as the "Primary Plot" in blue. This particular pattern is an actual (measured) BRDF, which happens to be substantially specular, or mirror-like. As such it peaks up at 0^o, indicating the highest reflectivity when the observation angle equals the incidence angle. It looks as if the pattern is fairly symmetrical in both angular directions, and it's first lobe reaches a minimum at about 25^o, where the first sidelobe begins to gain amplitude. This differs from typical antenna gain patterns in that the first sidelobe has the same amplitude sign (+ or -) as the main lobe. However, our model does not care, since it is an empirical one, and we can set the lobes to be of any sign we choose.

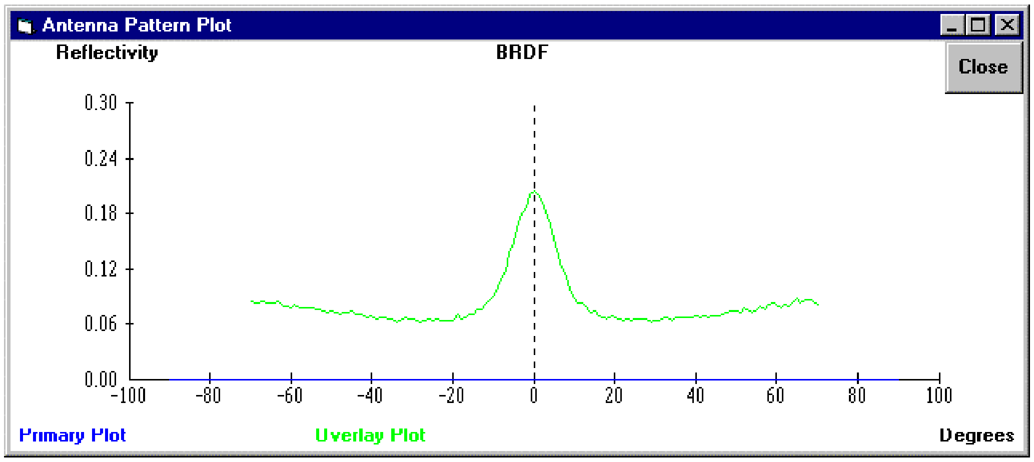
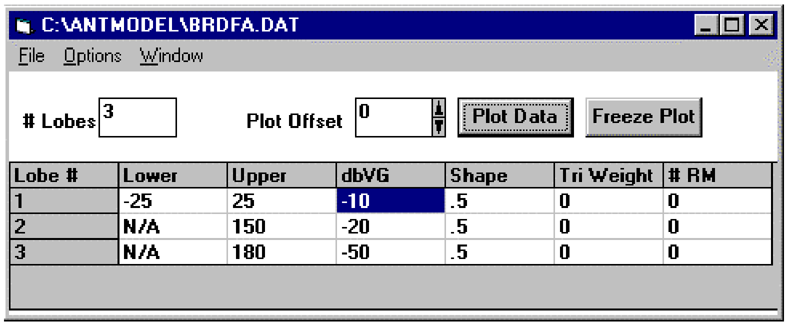


Figure 6. Actual BRDF to be Modeled

For a first attempt at a fit, we define some coefficients for our lobes in another ANTPAT screen, as shown in Figure 7. Here we can set the "Upper" and "Lower" lobe angular boundaries (γ_L and γ_U), our amplitude "dBVG" (M, the dB voltage gain), and the "Shape" (C, the shaping coefficient). We can also add "Tri Weight" (W, the triangular weighting coefficient), and can also add more lobes than the default number (3). The RM column allows radial modulation of sidelobe rings for non-uniform sidelobes, but we do not address that feature here.



# Lobes 3 Plot Offset 0

Lobe #	Lower	Upper	dbVG	Shape	Tri Weight	# RM
1	-25	25	-10	.5	0	0
2	N/A	150	-20	.5	0	0
3	N/A	180	-50	.5	0	0

Figure 7. First Fit Attempt Coefficients

Getting back to our example, we set the main lobe (Lobe # 1) amplitude to -10 (since our peak amplitude is less than 1, we know its a negative logarithmic quantity), and its upper and lower boundaries to 25^{o}, based on our earlier appraisal of these boundaries. We more or less arbitrarily set the first sidelobe (Lobe #2) upper boundary to 150^{o} (its lower boundary is automatically the upper boundary of Lobe # 1) and its amplitude to -20, since we know it is less than Lobe #1. Since we do not need three lobes, we just leave Lobe #3 set at its default values. We arbitrarily set all lobe shaping coefficients to 0.5, since we know from experience most lobes that have this kind of shape seem to need values around 0.5. Then we hit the Plot Data button and see what we have created, as shown in Figure 8.

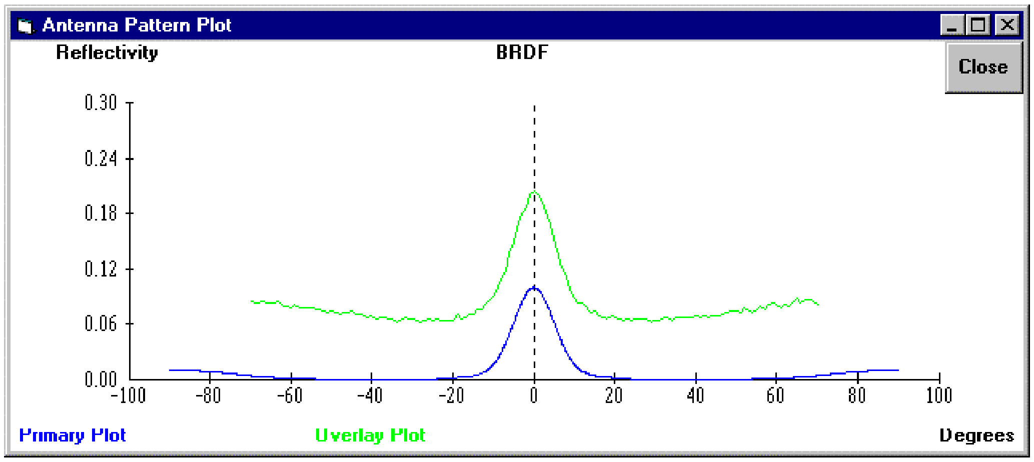


Figure 8. Results of the First Fit Attempt

Actually, this is not an altogether bad fit for a first attempt. It looks like we have a nominally decent main lobe, but the whole pattern is offset from the actual one, since it does not rest upon an amplitude value of 0.0, as our model does (it approaches zero at each lobe intersection). But this is no problem, as we can add some offset bias in ANTPAT. We do this first to get a better look at the main lobe fit we have achieved, adding an offset of 0.065 and obtaining the results Figure 9 illustrates.

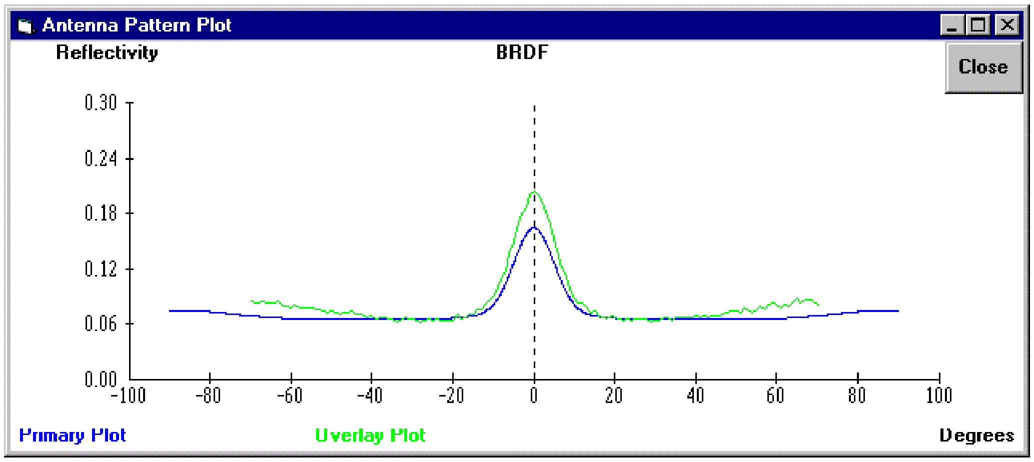


Figure 9. Results of adding some Offset Bias

Now the fitted pattern is meeting the actual one at +/- 25^{o}, and it looks like we need a little work on our amplitude and shaping coefficients. After a few iterations, we arrive at values of -8.6 for the amplitude and 0.45 for the shaping. This gives the results shown in Figure 10.

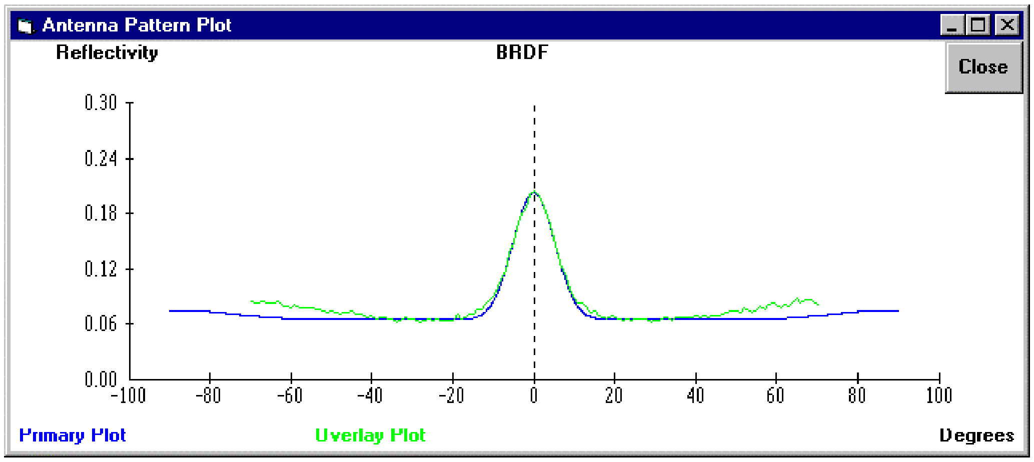


Figure 10. Results from Adjusting the Main Lobe Shape and Amplitude

Now this is a reasonable fit, especially at the main lobe peak, where the two plots are essentially concurrent, save for the noise upon the actual BRDF. In many applications, this would be good enough. However, we note that between about 10^o and 25^o, at the base of the main lobe, the fitted pattern is diverging a bit. Here is where the triangular weighting function comes in handy. We experiment a little and arrive at a triangular weighting coefficient of 0.09, which gives us the results of Figure 11.

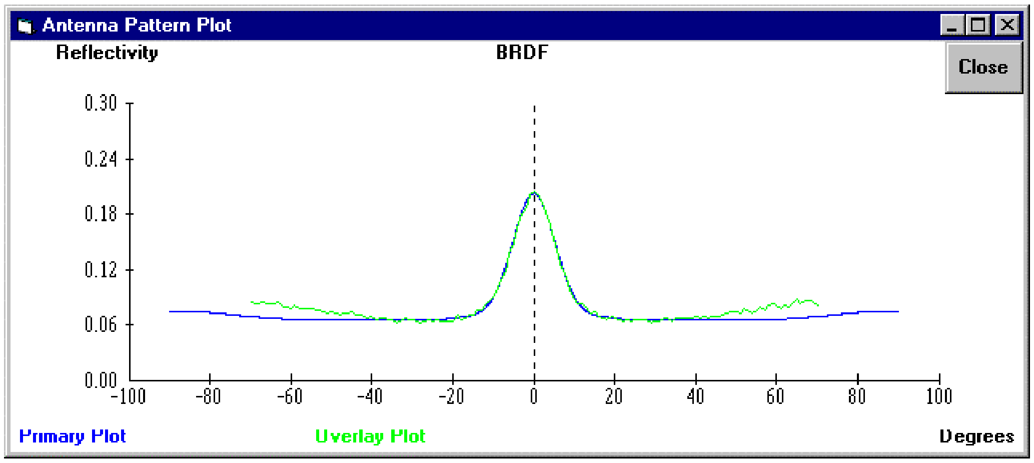


Figure 11. Optimizing the Main Lobe with some Triangular Weighting

That small adjustment certainly put the main lobe in great shape. Now we can work on the first sidelobe (Lobe #2). Recall that we arbitrarily set its upper angular boundary to 150^o. It looks like we need to bring that upper boundary down somewhat, to get this lobe's peak to occur earlier, and we also need to adjust the amplitude. Once again experimenting and iterating, we arrive at a Lobe #2 amplitude of -17 and an upper boundary of 130^o. Figure 12 illustrates the results of these adjustments.

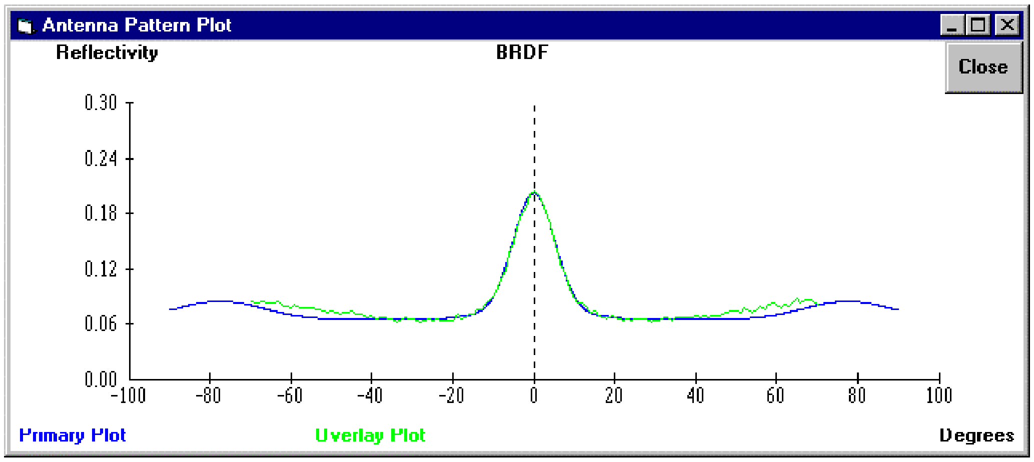


Figure 12. Results from Adjusting the Second Lobe Amplitude and Upper Angular Boundary

That seems to have placed the second lobe amplitude where it ought to be. Now it appears that our last chore is to adjust the second lobe shaping, to get the "dip," at around 60^{o}, out of the function. Again, some experimentation brought us to a value of 0.1 for the second lobe shaping coefficient. This produced the final pattern shown in Figure 13, whose aggregate coefficients we show in Figure 14. The fit is well within the noise characteristic of the actual BRDF.

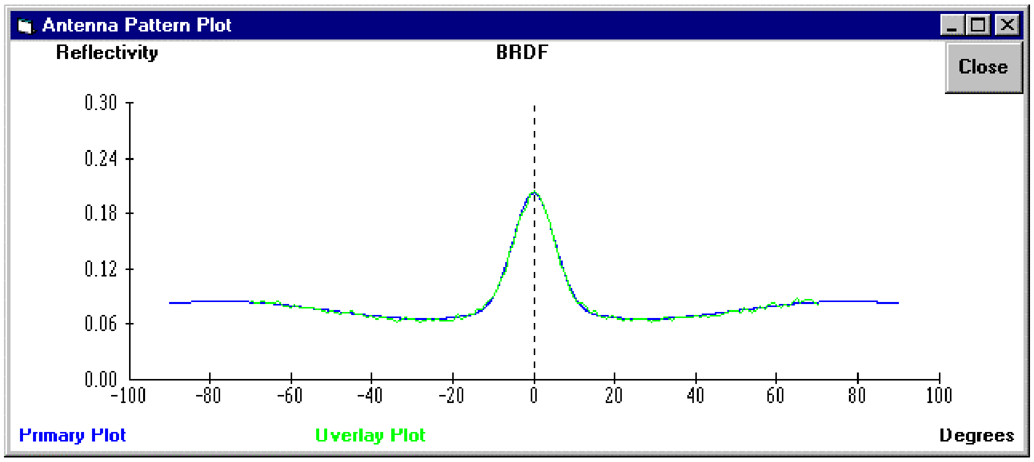
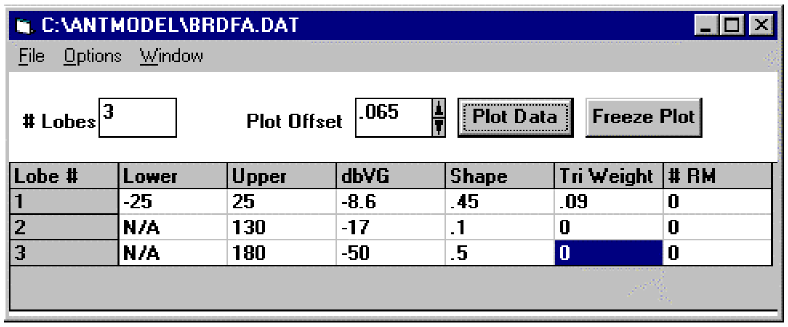


Figure 13. The Final Fitted Pattern



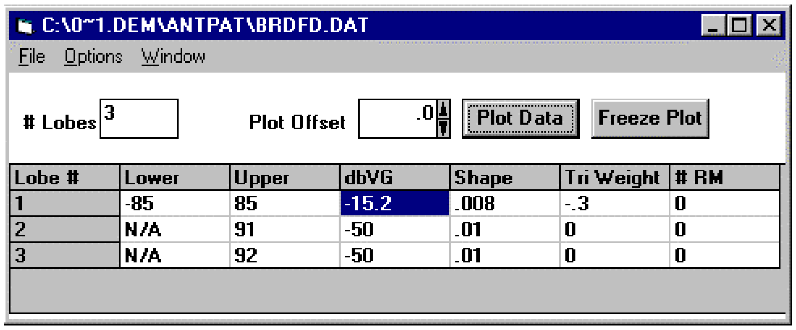
Lobes 3

Plot Offset .065

Lobe #	Lower	Upper	dbVG	Shape	Tri Weight	# RM
1	-25	25	-8.6	.45	.09	0
2	N/A	130	-17	.1	0	0
3	N/A	180	-50	.5	0	0

Figure 14. The Final Coefficient Set

A second example pattern is another BRDF, with a somewhat different reflectivity characteristic than the previous one. Figure 15 illustrates the coefficients used to configure RIPS in the ANTPAT Tool, to create this pattern. Note that here we use significant triangular weighting, to create a "dip" at the angular center of the single lobe. The entire BRDF is described using three lobes and twelve coefficients.



# Lobes 3   Plot Offset .0

Lobe #	Lower	Upper	dbVG	Shape	Tri Weight	# RM
1	-85	85	-15.2	.008	-.3	0
2	N/A	91	-50	.01	0	0
3	N/A	92	-50	.01	0	0

Figure 15. ANTPAT Coefficients for a Second BRDF Example.

In Figure 16, we show the RIPS-produced pattern fit to this BRDF pattern. Similar to the preceding example, this BRDF pattern was a measured one, and not a mathematically-calculated one. Hence, we once again have some noise upon the actual pattern data. Should we desire to simulate a noise characteristic, the amplitudes produced by RIPS for the various angles may easily be modulated with an appropriate noise distribution function. In the actual BRDF, there appears to be an anomaly at about -30^{o}. This is possibly a flaw in the measurement or recording process, as the data goes to exactly 0.0 here. If, however, we need to model dead spots or anomalies such as this one, we can blank or modulate selected angular regions any way we choose. Again, we see that RIPS provides an excellent, fast-executing modeling capability. It simply matches the lobe shapes, and can handle noise and anomalies as we desire.

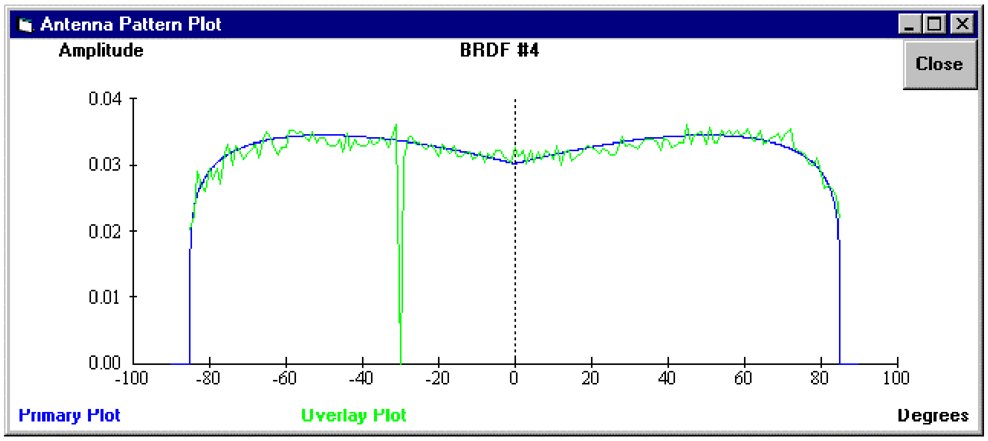


Figure 16. ANTPAT-Generated Pattern Fitted to Actual BRDF Overlay.

As Figures 13 and 16 illustrate, in both of the example cases we presented the final fit is quite good, well within the noise characteristic of the actual patterns. We routinely and easily achieve such fits to any electromagnetic interference patterns, using the simple, iterative process we have just described. We basically fit the main lobe first, and move outward, fitting each subsequent lobe, one at a time. Since each lobe's function is continuous only within its borders, we never have to worry about some sidelobe adjustment affecting another lobe. In practice, we have found that even highly-complex antenna patterns, some containing dozens of side lobes, can be fitted in less than an hour by someone practiced in the use of this method. Many patterns are fitted in five minutes or less. Then, in a real-time simulation host, the coefficients are simply stored to support function calls to this method, interrogated by the angular position locating the applicable coefficient set, as we explained earlier.

5.0 SUMMARY AND CONCLUSIONS

In summary, we have found the algorithms and methods described herein to be practical, useful, and valuable, across many instances of BRDF simulation or other electromagnetic interference modeling applications. Once a user becomes accustomed to the control over the lobes provided by the algorithm coefficients, they are able to quickly and easily create the desired pattern. We see the value of RIPS as providing an accurate, fast-executing method for simulating all types of electromagnetic interference phenomena, with minimal storage requirements. It offers the accuracy of a rigorous model or high-resolution lookup table, at a fraction of the execution time or storage requirements. We know it is accurate, and that it will execute quickly, due to its extreme simplicity. What it requires is a system for fitting the model to an actual pattern, to obtain the coefficients needed for real-time execution.

MTL Systems, Inc. is currently looking into providing a software toolkit for using this technique, with supporting file management utilities and tools to both mathematically and graphically assess the quality of the fit a user is achieving. These additional embellishments will render RIPS a useful tool for modelers to capture and apply the patterns they need for myriad simulation requirements. We foresee such an implementation as shown in Figure 17, where the “core” pattern simulation algorithms of RIPS (as it exists now) are complemented with a *Create & Capture Tool* and a *Compare Tool*.

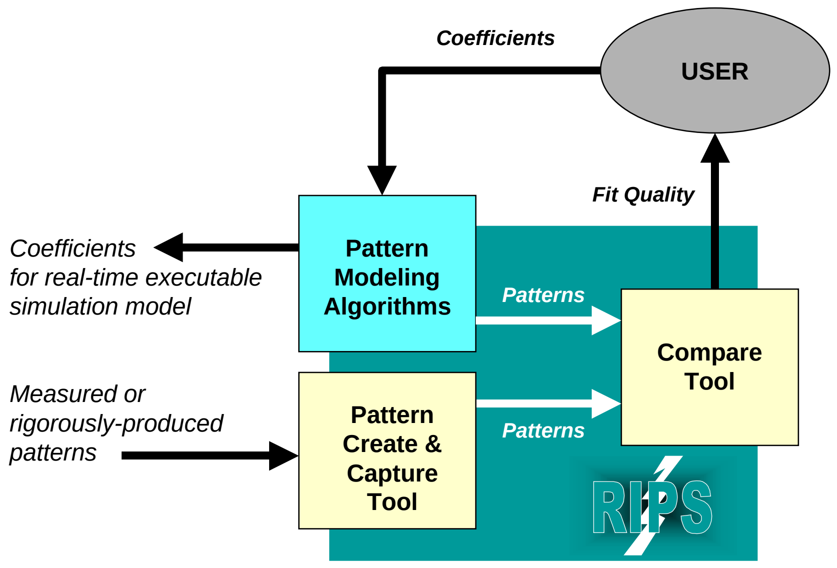

Coefficients
USER
Fit Quality
Coefficients for real-time executable simulation model
Pattern Modeling Algorithms
Patterns
Compare Tool
Measured or rigorously-produced patterns
Pattern Create & Capture Tool
Patterns
RIPS

Figure 17. The RIPS Tool Concept – CAD for EM Patterns.

This system would provide automated and user-friendly access to the fundamental algorithms and the new utilities. The Create & Capture Tool would provide the means to import existing patterns, either from measured sources or from rigorous mathematical expressions. This tool would also have its own rigorous pattern modeling algorithm library for producing patterns. It would then translate or format these pattern data for use by the Compare Tool. The Compare Tool would provide a complete pattern visualization and comparison capability. Using this tool, the user would be able to take patterns from the Create & Capture Tool, view them from any desired aspect, and compare them with the patterns created by the Algorithms. The user could then iteratively adjust the coefficients for the Algorithms and, once satisfied with the fit, export them for use by a real-time implementation of the Algorithms in a target simulation.

In summary, our concept is to take the existing pattern simulation algorithms and embed them within a system for easily creating and verifying simulated patterns for subsequent real-time simulation needs. It is, in essence, “CAD for EM Patterns” and would be a significant asset for real-time or distributed collaborative enterprise simulation, for IRCM/IRCCM assessment and many other applications.

Reference

[1] Hirsch, Herbert L., and Douglas C. Grove. *Practical Simulation of Radar Antennas and Radomes*. Artech House. Boston. 1986.